

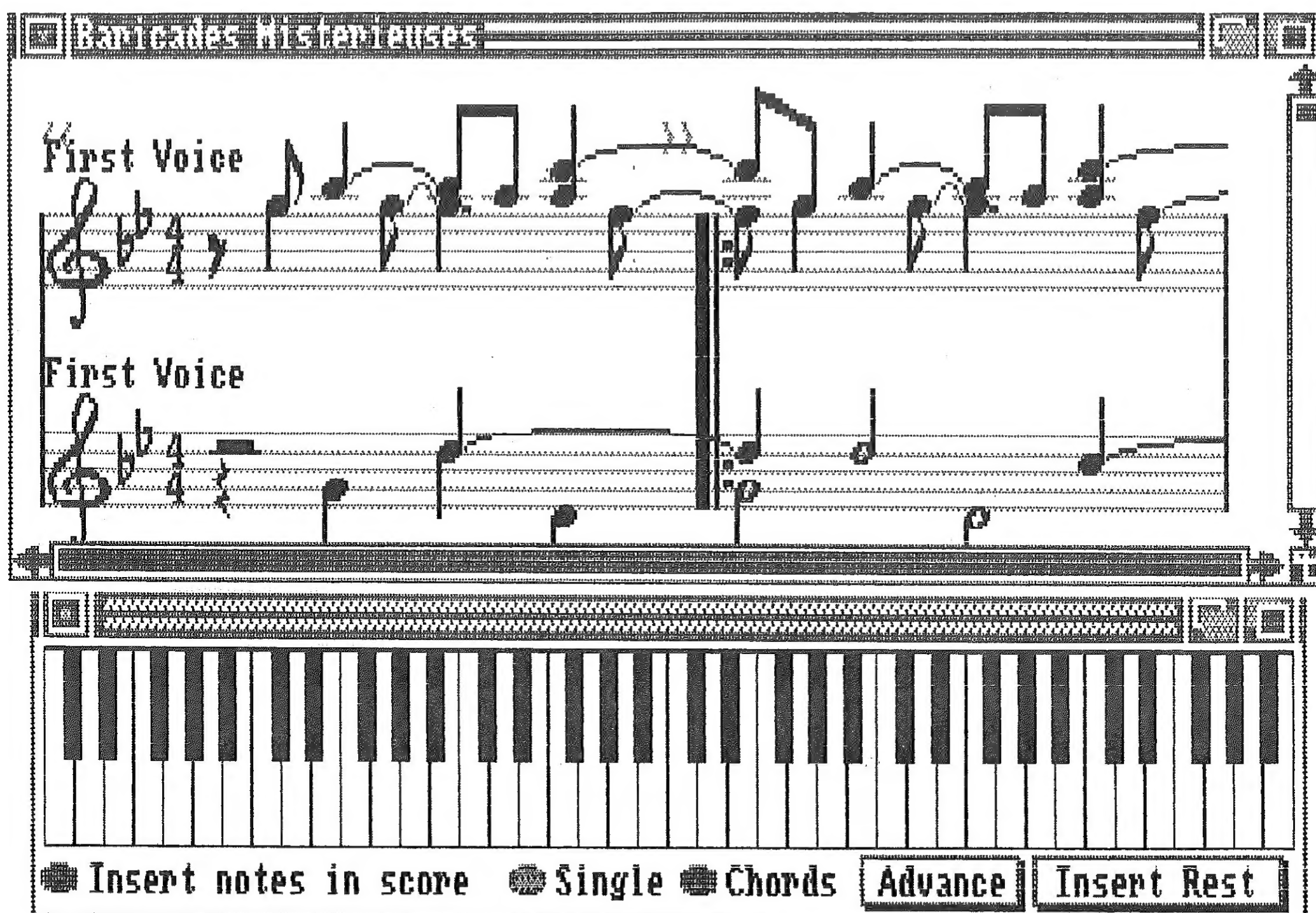
WORKBENCH

Registered by Australia Post — Publication No. VBG7930

Number 12

Circulation: 650

May 1987



Next Meeting

Demonstration of Music and the Amiga

Sunday, May 14th, 1987 at 2pm

AUG meetings are held at Victoria College, Burwood Campus
in Lecture Theatre 2. Melways map 61 reference B5.

Amiga Users Group, PO Box 48, Boronia, 3155, Victoria, Australia

AMIGA is a trademark of Commodore—Amiga, Inc
The Amiga User's Group has no affiliation with Commodore

AMIGATM Users Group

P.O. Box 48, Boronia, 3155, Victoria, Australia

Amiga Users Group

The **Amiga Users Group** is a non-profit, self-help group, made up of people interested in the Amiga computer and related topics.

Club Meetings

Club meetings are held at 2pm on the second Sunday of each month at Victoria College, Burwood Campus, in Lecture Theatre 2. Details on how to get there are on the back cover of this newsletter. The dates of the next few meetings are:

Sunday, May 10th at 2pm (Mother's Day)
Sunday, June 14th at 2pm
Sunday, July 12th at 2pm

Production Credits

This month's **Amiga Workbench** was edited by Peter Jetson. Equipment and software used was: TurboDOS S-100 computer, Diablo 630 printer, Gemini 10x printer, Wordstar, Fancy Font and Grabbit.

Copyright and Reprint Privileges

This entire journal is Copyright 1987 by the Amiga Users Group. Articles herein that are explicitly marked as having restricted reproduction rights may not be reprinted or copied without written permission from the **Amiga Users Group** or the authors. All other articles may be reprinted for any non-commercial purpose if accompanied by a credit line including the original author's name and the words "Reprinted from **Amiga Workbench**, newsletter of the **Amiga Users Group**, PO Box 48, Boronia, 3155", and a copy of the publication in which the reprint occurs is sent to us.

Contributions

Articles, papers, letters, drawings and cartoons are actively sought for publication in **Amiga Workbench**. It would be appreciated if contributions were submitted on disk, since that means they don't have to be re-typed! We have access to a wide range of computers, so we should be able to accept almost any type of disk, but Amiga disks are certainly the easiest. All disks will be returned! Please save your article in **text-only** format. Absolute deadline for articles is 16 days before the meeting date. Contributions can be sent to:

The Editor, AUG, PO Box 48, Boronia, 3155

AUG Users Group Disks

Disks from the **Amiga Users Group Library** are available on quality 3.5" disks for \$10 each including postage on AUG supplied disks, or \$2 each on your own disks. We can also provide 80 track 5.25" Amiga format to special order. Please enquire. The group currently holds 69 public domain volumes, mostly sourced from the USA, with more on the way each month. 19 extra volumes should have arrived by the time you read this.

Member's Discounts

The **Amiga Users Group** is currently negotiating discounts for its members on hardware, software and books. Members will be notified when negotiations are complete.

Currently, **Technical Books** in Swanston Street in the city offers **AUG** members a 10% discount on computer related books, as does **McGills** in Elizabeth Street. Just show your membership card. Although we have no formal arrangements with other companies yet, most seem willing to offer a discount to **AUG** members. It always pays to ask!

Membership and Subscriptions

Membership of the **Amiga Users Group** is available for an annual fee of \$20. To become a member of **AUG**, fill in the membership form in this issue (or a photocopy of it), and send it with a cheque for \$20 to:

Amiga Users Group, PO Box 48, Boronia, 3155

Amiga Users Group Committee

John Holland . . .	(Co-ordinator) . . .	348 1358	Carlton
Bob Scarfe	(Vice Co-ordinator) .	376 4143	Kensington
Ron Wail	(Meeting chairman) .	878 8428	Blackburn
Eric Salter	(Secretary)	861 9117	Kew
Paul Radford . . .	(Treasurer)	663 3951	BH only
Neil Murray	(Membership)	792 9666	Dandenong
Bohdan Ferens . . .	(Purchasing)	792 1138	Dandenong
Geoff Wood	(Book Librarian) . . .	580 7463	Aspendale
Geoff Sheil	(Software Librarian) .	509 3151	Armadale
Ron Wail	(Software Librarian) .	878 8428	Blackburn
Peter Jetson	(Newsletter Editor) .	762 1386	Boronia
Stephen Thomas	830 5783	Canterbury
Jo Santamaria	836 9129	Canterbury
Ron Van Schyndel	882 7264	Hawthorn
Mike Creek	878 9039	Blackburn

When phoning committee members, please try to be a bit considerate and not call at meal-times, late at night, or during popular TV programs. If you only have a general query, try to ring the member who lives closest to you.

Textcraft Tips by Mark Kelly

After reading a few patronising reviews of Textcraft ("It's a cute little attempt" style of comment), I decided to wait for the advent of Vizawrite before launching into a word-processing career.

Unfortunately, Vizawrite and I are waiting for the official release of Kickstart V1.2, so I invested in Textcraft and promptly wrote a novel on it with few regrets of difficulties. [Editor's note: Vizawrite (at last) looks like being released about the end of May.] Textcraft was good, but the quality of the manuscript is open to debate. You might be interested to know that the 210 page novel fitted onto a single disk! If you are a desktop publisher, a power freak, or you can't spell your own name with a Macquarie dictionary nearby, then you would be better to wait for a more sophisticated system, but for basic text entry, editing and printing, Textcraft is a good choice.

Bugs - One early problem that had me infuriated at first was system crashes, until I realised that I had been using my Beta release of Kickstart V1.2 obtained by being nice to a Commodore dealer.

Rebooting with version 1.1 cured that problem. Whether or not Textcraft's aversion to Kickstart V1.2 will be cured in the official release will be interesting to see. Just don't scrub your old Kickstart V1.1 too soon after getting the new version! There remains a perplexing habit of Textcraft of getting confused when opening a document from disk. After opening a few documents in turn, Textcraft decides to start hiding files. Eventually, the list of files it pops up in its requester box is completely empty! The only solution I found was to quit and restart. Another bug appears to be somewhere in the SET PAGE NUMBER AND TITLES command which will occasionally summon the guru. A similar occasional glitch appears when printing - after successfully printing several documents, the system would appear to forget where it put its printer driver and try finding it on the Textcraft disk, give up in disgust and sit twiddling its silicon thumbs. As an old-time TRS-80 and Hitachi Peach hacker, I tend to regard such idiosyncratic software quirks as normal. Maybe I'm being too soft-hearted with an international company that should be more careful with its products, but once you learn Textcraft's ways, it is a friendly mule to work with. If you buy Textcraft expecting it to be Black Beauty, you'll be annoyed.

Wish List - Since I have no need of a spelling checker (preen, preen), mail-merge facilities, kerning (whatever that is) or snaked columns, my wish list is modest. I really wish Commodore had organised Textcraft's document printing to be a background task so I could do something more useful while an umpteen page document churns from the dot matrix and I make yet another cup of coffee and watch half of Blues Brothers on video. One would think it would've been easy to do two things at once on a super-whiz-bang multi-tasking multi-processor computer! If I had an extra one or two megs I could run two copies of Textcraft at once, I suppose, with one printing and the other editing the next chapter, but memory expansion is a little too expensive still.

It would also be nice to be able to cut and paste between documents, but if one is desperate, it can be done under CLI. Use the Join command to merge two files that must have been saved in TEXT ONLY format. (ie JOIN DOC2 DOC1 AS BIGDOC). Invoke Textcraft with the name of the resultant file (DF0:TEXTCRAFT DF1:BIGDOC), cut out the unwanted parts and paste the desired parts into position, then save the new improved document. Be warned that this is not guaranteed to

work. Sometimes it does, sometimes it doesn't, so **never** use your original disk or your only copy of a three-hundred page masterpiece to experiment with. Good luck.

One of Textcraft's omissions is a word-counting utility which I need as an author. Below, you will find a basic program that will count words in a standard Textcraft document (ie it does not have to be a "TEXT ONLY" version). Word counting programs are necessarily inaccurate to some degree, but this one is quite reliable and (more importantly) fast. Note in the fourth line a little trick I thought up to ensure the use makes the window active before trying to type into it (and getting beeped and flashed).

Another tip for those of you who wish Textcraft had some sort of macro keystroke ability (don't you love jargon?) so one key could produce a long, commonly repeated keystroke sequence. One could then hit a key or two instead of typing in "anthropomorphically" a dozen times. This feature is commonly handy in producing a long document when one finds oneself typing the characters' names seven hundred times, or with words that one **always** mistypes no matter how careful one is (I have a particular aversion to "through" - for some reason I insist on putting a "t" on the end every time!). Textcraft's FIND AND REPLACE can come to the rescue with a poor man's "Hotkey" feature.

Simply replace the offending word or phrase with a unique keystroke and another key. I use the tilde (the SHIFTed version of the key under the ESC key) which no-one uses in normal writing as my unique key. As I typed my novel, for example, I just replaced the name Elizabeth with "e (tilde + e). When the chapter was finished, I just called up FIND and REPLACE and replaced "e with Elizabeth. The other characters' names were replaced similarly with "b, "m, "n, "t and so on. Ensure that each abbreviation is unique - problems may arise if you used something like "b for Bill and "bo for Bob. If you replaced "b with "Bill", you'll end up with lots of "Billo"s appearing! Use it cautiously and it can be very useful. I hope these hints prove very helpful.

WIDTH 70

```
PRINT "COUNTS WORDS IN TEXTCRAFT DOCUMENTS"
COLOR 3 : PRINT "Click in this window!" : COLOR 1
WHILE MOUSE(0) = 0 : WEND 'Wait for window to become active
```

```
PRINT "Document disk should be in DF1:"
COLOR 3
INPUT "Enter PATH name DF1:", inam$ : COLOR 1
inam$ = "df1:" + inam$
```

```
OPEN "i", #1, inam$, 20000 : w = 0
```

```
WHILE NOT EOF(1)
    LINE INPUT #1, a$ : pointer = 1 : PRINT a$
    IF LEN(a$) > 7 then
        WHILE pointer <> 0
            pointer = INSTR(pointer + 2, a$, " ")
            count = count + 1
        WEND
    END IF
WEND
PRINT "Total" count "words"
CLOSE
```

Introduction to AmigaDOS - Part 4

This month we look at batch processing and background jobs. While not strictly batch procesing, the EXECUTE command, in combination with a command file (which is interpreted by EXECUTE), allows us to do some very nice programming. Why would you ever want to use EXECUTE? Try compiling a C program without it! An EXECUTE command file, allows us to substitute parameters which are then used within the command file as if we typed them there ourselves. For example the MAKE file in lettuce C allows us to compile our C source file into object code. This is done by typing:

EXECUTE MAKE [path]foo.c

This will allow us to compile a source file anywhere on the amiga file subsystem. The actual file MAKE is an EXECUTE file and allows multiple parameter substitutions for specifying libraries to search etc. To compile the above program by specifying it all to the compiler on the command line takes just over 120 or so characters. If you get one character wrong or miss one / in specifying directory search paths, then you're cactus basically! The whole process generates mountains of compiler diarrhoea which cannot be stopped at best, or generates subtle bugs which will not be apparent until link time or worse, run time (very rare though).

The EXECUTE command is supported by a galaxy of commands that are useful for altering the flow of control inside a command file. These are: IF, SKIP, FAILAT, LAB, ECHO, RUN, QUIT.

Basically, what happens when you invoke EXECUTE, is that EXECUTE is loaded. The CLI searches for the file you specified on its command line. If this is found, the CLI does funny things to its CLI process by inserting a BSTR string [this is gory tech detail and is probably uninteresting] containing the name of the EXECUTE command file in the structure pointed to by the CLIStruct field of the AmigaDOS process structure. If the first line of your command file contains a . DOT directive, then the CLI builds a temporary file in the T: directory. "Yeah, so what?" you say. The upshot of this is that the CLI now accepts command lines from the EXECUTE file instead of the console keyboard.

The command file may contain text other than the invocation of commands. These are called directives and like compiler directives, are the property of EXECUTE rather than commands that are searched for on disk. They direct EXECUTE to do certain things:

- . The dot directive, indicates that a compiler directive is following. It is really a directive introducer and has no other effects. It may be changed with the DOT directive.
- DOT The DOT directive .DOT ch, changes the . directive introducer to the ch character where ch is any character.
- BRA The .BRA directive, is used to re-define the left bracket symbol from its default of <. This character is used to enclose formal parameter arguments. If you want your formal arguments to contain the < character, you can re-define BRA so you can use <.
- KET Similar to BRA but re-defines the > character, <s better half!
- DOLLAR This command re-defines the \$ character to any other. You use the \$ to separate your formal parameter from its default (if a parameter is not specified on the command line, it is replaced by the default).

KEY This directive tells EXECUTE how many parameters to expect on the command line, and what order to expect them in. E.g. .KEY progname,includefile,outputfile ; would inform EXECUTE to expect three parameters in the above order.

Now when we invoke the file we would type:

EXECUTE commandfile foo.c stdio.h myprog.o

This will invoke commandfile which will cause, for example, the file foo.c to be compiled using stdio.h as the include file generating myprog.o as the object file. If we change the order of the parameters on the command line we must specify which formal parameters we wish it to be associated with:

EXECUTE commandfile include stdio.h progname foo.c myprog.o

Here we have spelt out explicitly which of the parameters we intend to be associated. We did not specify this with myprog.o because after the interpreter has assigned values, any remaining input is used to fill the leftover keyword positions from left to right.

We can specify conditions for formal parameters by appending either a /a or /k to it. The /a append specifies that this parameter is required - no options! The /k append specifies that while the parameter is optional, it must be preceded by the explicit keyword. E.g. If outputfile/k was used, then if we specified an output file as in myprog.o, we must say: outputfile myprog.o.

Build Your Own 1 Megabyte RAM Card!

Features: 1 megabyte of RAM (using 41256's and the Texas Instruments THCT4502 controller), battery backed up clock (Motorola MC146818) and hard disk controller expansion port (compatible with the Western Digital WD1002-05 controller card). The board plugs directly onto the side of your amiga 1000. It does not autoconfigure (requires the 12 AddMem command) and does not pass the bus

For just \$95 (inc p&p) you get a blank, double sided PCB with plate through holes, solder masks and silkscreen legend, a pre-programmed PAL chip (20L10), schematic diagrams, parts list and a 3.5 inch floppy disk containing public domain software for driving the clock. No case or other parts are provided.

This project is only recommended for people with prior electronics experience. I am simply an individual trying to make relatively cheap Amiga expansions available. The estimated cost of total additional parts is \$350.

To order, see Alan Kent at the next user group meeting, or phone 232 8300 (weekends). This is a limited once only offer.

How do we use the parameter that we enter? Simple, we can now specify where we want to put them by enclosing the formal parameter in angle brackets [or their default]. E.g.:

COPY <inputfile> to <outputfile>

Here, whatever we supply as the actual parameters will be substituted between the < >. I.e. If we type:

EXECUTE commandfile df0:c/#? df1:

we would copy all the contents of the c directory of df0: to df1: root directory. Obviously this is a stupid example as it is simpler to use the COPY command on its own but it serves to illustrate a point.

Conditionals

We can use the IF command to alter execution flow based on tests we make.

IF <condition>	IF <condition>	IF <condition>
<command>	<command>	<command>
ENDIF	ELSE	IF <condition>
	<command>	<command>
	ENDIF	ENDIF
		ENDIF

Above are the three general forms of the IF statement. We make a test, if the condition is satisfied, we execute the command following, or we either default to the next command after the ENDIF terminator or there is an alternative with the ELSE. The third form illustrates that IF constructs can be nested but their extents cannot overlap. It is a good idea to use structured formatting to avoid confusion as to where tests end.

When a command executes, it will return a code to the CLI based on what happened during its execution. If it succeeded, usually it returns 0. We may make tests with IF based on that returned code:

IF WARN	; previous command returned code >=5
IF ERROR	; " " " " >=10
IF FAIL	; " " " " >=20

We can now take appropriate action based on these return codes. The command QUIT will allow us to terminate a control sequence and return a value with QUIT n. As command files can be nested, i.e. command files can invoke other command files, we can test the return codes generated by command files as they return control to the one [conceptually] above. The FAILAT command sets the level of return code that will cause a fatal termination of a command sequence. If code returns a code greater than the current FAILAT level [default 10] then the sequence will terminate.

Other tests that can be made with IF are:

IF <a> EQ 	; if the text of a and b are equal = true
IF EXISTS <file>	; True if the DOS can find the file.
IF <a> EQ ""	; Test for unset or NULL parameters.
IF NOT <condition>	; reverse the test for a condition!

Labels

We can specify labels for sections of code in the command file and we can continue execution at these places based on tests that we perform with IF. We define a label with the LAB command. We modify control flow with SKIP.

IF NOT EXISTS <file>

SKIP notfound

ENDIF

.

.

.

LAB notfound

ECHO "File specified does not exist. Execution failed"
QUIT 20

If it can't find the file it will execute from notfound and print the error message and quit!

A note on command files: Command files can call themselves i.e. are recursive. See the example in the DOS manual!

Background tasks

We can create a background task that executes concurrently with the CLI. The RUN command creates another CLI process but one which is non-interactive. This means it cannot accept input from the keyboard although programs that run under it may accept input from a file. RUN inherits the same stack and current directory of its parent CLI. We can RUN multiple commands by placing a + sign between commands. We can even run EXECUTE files in the background while doing other things with the RUN EXECUTE <filename> construct.

We have covered a little of the power of EXECUTE files. Next time we may delve a little deeper into the innards of AmigaDOS.

-- Eric Salter

FREE NEWSLETTER

AMIGA NEWS

and

Software Information

(ALSO WHAT'S UNDER DEVELOPMENT)

(Local and Overseas)

and

SPECIALS ON SOFTWARE

CATALOGUE INCLUDED

MAXWELL (03 419-6811)

162 Nicholson Street, Abbotsford, Vic 3067

AMIGA SALES/SERVICE/HIRE

Send to: FREEPOST 2 (No Postage Required)

MAXWELL

162 Nicholson Street
Abbotsford, Vic 3067

Name: _____

Address: _____

Using Workbench Facilities in your Programs

Workbench is the interface I'm sure you are all familiar with on the Amiga - at least from a users point of view. It allows interaction with the Amiga filing system and applications by using icons and the mouse. From the programmers point of view, Workbench provides a number of library functions to allow relatively easy manipulation of an application's icons. In this article, I'll endeavour to explain how to generate icons and how to have your program use them at startup. My apologies to those of you who are not familiar with the C programming language, but the code fragments used in this article and the demonstration programs are both in C.

This article will be, by necessity, of a technical nature. Please persevere, I'll try explain most terms I use and even if you don't write software, you'll learn how to use the workbench interface when starting programs (provided of course those programs use the facilities available). To accompany this article I've written three programs (available on an AUG Public Domain disk) and the following description of these also details what I'll be talking about:

icon2c.c Uses an icon created with IconEd (in the system drawer of Workbench) and generates the C source for the DiskObject structure including the image and Gadget structures (these define what the icon will look like as well as other info explained below).

write_icon.c Shows how to use the C source generated by icon2c.c to generate an icon associated with each file an application generates. The icon includes information which allows its use with the next demo program. One of the icons has a separate image when selected.

wb_startup.c Demonstrates the code necessary (in addition to that in Astartup.obj which handles the common startup and exit tasks) to handle both a CLI and a Workbench startup. When started from workbench, the user can preselect a number of related project files, double clicking on the last one and the program will be automatically loaded. It then interprets the startup message workbench sends and loads the preselected projects. To make the demo a little more interesting the following Intuition features are also shown:

- opening a custom hi-res screen with its own window and setting the 8 colours for that screen.
- how to direct the system requesters to appear in your screen instead of being hidden behind on the workbench screen.
- use of the auto requester feature for easy generation of your own requester.

The above programs are written solely using Amiga library calls and are compiled without the Lattice library LC.LIB, allowing quite small executable files to be produced.

More About Workbench

The Workbench facilities that I will describe are the ability for the user to preselect any number of icons that are related to an application, double clicking on the last data file icon and automatically having the application program executed and that program then loading each of the data files with the ability to determine the type of file (irrespective of its file name) and use it in the appropriate way. The application can also determine if the files belong to it or not. The application needn't be in the same directory as the data files. For instance if you're not sure which word processor was used to create a document then by double clicking the document the word processor should execute and load the document. Of course this won't happen unless the application program properly uses the Workbench interface.

using the code:

```
IconBase = OpenLibrary(ICONNAME,0)
```

This causes the library to be loaded (this library is normally disk resident in directory "DiskName:lib") and the address returned in IconBase. ICONNAME is defined as "icon.library" in the include file workbench/icon.h.

The Gadget Structure

The gadget structure do Gadget is used to hold the icon's image. This structure is defined in intuition/intuition.h. The important members of this structure for an icon are:

- Width & Height - the width and height in pixels of the icon active region for selection with the mouse.
- Flags - defines the type of alternate image displayed when the icon is selected.
- Activation - should have RELVERIFY and GADGIMMEDIATE set. This causes the icon to show the selected image (may be just complemented colours) when selected by the mouse but the icon is not activated unless the button is released while still over the icon.
- Type - should be BOOLGADGET (means boolean gadget)
- GadgetRender - points to the icon image structure normally displayed.
- SelectRender - if GADGIMMEDIATE is set in Flags this should point to the image structure that will be displayed when the icon is selected.

Generation of the DiskObject Structure

It's a fairly simple matter to generate the C code for this structure with the exception of the image structure (or structures if you are supplying a select image) which requires the pixel information in two planes. As the Icon Editor supplied on the Workbench disk generates the DiskObject structure for any icon you design, this offers an easy way of generating the C code for the image structure. The following C code fragment loads the icon, "filename.info" into memory and provides a pointer to the image structure:

```
struct DiskObject *diskobj;
struct Image *do_image;

diskobj = (struct DiskObject *)GetDiskObject(filename);
do_image = (struct Image *)diskobj->do_Gadget.GadgetRender;
```

Now it's just a matter of extracting the relevant members of the image structure (Width, Height, Depth (always 2 for an icon) & ImageData). The ImageData is required as two planes of words (16 bits/word).

My program icon2c.c generates the code for the image structure and the remainder of the DiskObject structure. The C source output can be modified to suit your needs. Design your icon image with IconEd and then run icon2c on the .info file created by IconEd and you have the C source for the DiskObject structure.

ToolTypes Array

This is part of the DiskObject structure (do ToolTypes) and is treated separately as it provides for user information. This member points to an array of free format strings (each string can be up to 32Kbytes long but it should not contain a linefeed, i.e. should not be over one line long). The convention encouraged by Workbench for each string is:

```
<name>=<value>[|<value>]
```

where <name> is the field name and <value> is the text associated with that name. Multiple values are separated by a vertical bar. The strings used in my example program write_icon.c are:

```
FILETYPE=sample_code.example|text (for text file)
FILETYPE=sample_code.config|text (for config file)
```

the "sample_code.example" and "sample_code.config" values are used to identify the file types that the icons represent. The tool activated recognizes these strings as its own files.

The "text" value is to indicate that the files represented by the icons can also be used by a tool that handles text files (e.g. an editor or word processor). Use of this convention allows the use of two system routines:

FindToolType(diskobj->ToolTypes,"FILETYPE") returns the value of a ToolType element, for the config icon "sample_code.config|text" would be returned.

MatchToolValue(value,string) returns true if string is in value. In our case this routine will return true for a string of "sample_code.config" or a string of "text" as it knows how to parse the vertical bars between values.

The above routines are more relevant when a tool is evaluating the startup message sent to it when it is activated - I'll spend more time on this later, first we have to generate icons that the user can select!

Generating Icons

As I explained at the beginning of this article (for those of you who can remember back that far) to use the Workbench interface the tool must generate an icon for each file it saves. So far we've seen what an icon consists of and how to generate the C code for the DiskObject structure. The next part's really quite simple. The C fragment to do this is:

```
struct DiskObj *diskobj; /* points to the structure
                           generated by icon2c */
char *filename; /* points to filename (without the
                  .info) */
int status;

status = PutDiskObject(filename,diskobj);
```

status is true (non zero) if the icon file was created. For the full code to generate two different icons, see the example write_icon.c. Normally the tool would generate the icons with each file it creates as well as handle the startup selections. For demonstration purposes I've separated these functions in the demo to avoid confusing the two issues. Now on the final section which ironically is one of the first things a tool does when it is activated.

Program (Tool) Startup

A tool can be started by typing its name on a CLI command line. CLI allows the tool name to be followed by arguments which the tool can evaluate. These can be file names or whatever the tool expects to see. In the case of file names, to tell the difference between different file types the file names must be coded (e.g. file.txt or file.con) or the file type imbedded at the front of the file. A tool can also be started by activating its icon or one of its project icons (data files). In this case the icons selected at startup are passed to the tool as a Workbench Startup message. By evaluating the icons preselected (using the ToolTypes array) the tool can determine what type of files the user preselected and if they were generated by this tool or if they are invalid and should be ignored.

The environment for a tool started under Workbench is quite different to that when the tool is run from the CLI. Workbench runs the tool as a separate process, running asynchronously to Workbench (i.e. multiple tools can be started from Workbench and run at the same time :- multitasking). The CLI does not create a new process for the tool, it jumps to the tool's code and the tool shares the process with the CLI. One of the things that the tool has access to is the CLI file handles for input and output. By default a program started from Workbench does not have a window or valid file handles for input and output (stdin, stdout). If the tool attempts to printf() without these it will crash the system. As soon as the tool has been started Workbench sends it a startup message containing the environment for the tool.

The actual program startup is handled by standard startup code which is linked in before the program code (AStartup.obj if you link with the Amiga library, Amiga.lib, or LStartup.obj if you link with the Lattice library, lc.lib). This code waits for the startup message, opens the tool window if one has been requested in the icon do ToolWindow member,

Providing the user of your applications with the full facilities of the Workbench interface is a two edged sword. Before the facilities can be utilized the application must generate icons for each project (data file) that it creates (I'll explain these terms in a minute). These icons must include not only the image information but also the information necessary to tell workbench which application (default tool) to load when the project icon is double clicked, the stack size for running the program, the type of icon (tool, project etc.) and any miscellaneous information necessary in the ToolTypes array. To see this information for a typical icon, single click an icon and then select Info in the Project menu of Workbench. This will display most of the above for that icon (documents created by textcraft include the ToolTypes array and default tool information).

Terminology

The terms used with the workbench interface are fully explained in the ROM Kernal Reference Manual: Libraries and Devices Chapter 18 (That's the THICK manual). Here's a summary of what's in Chapter 18:

- Tool - An application program or system utility.
- Project - A file produced by an program and associated with that program, e.g. a text file, drawing etc.
- Drawer - A disk based directory.
- Icon - This is shorthand for a Workbench Object, it can be in memory or on a disk.
- Workbench Object - contains all the information that Workbench needs to display and use a project, tool or drawer. The type of Workbench object that we will be discussing is the DiskObject.
- info file - A disk representation of an icon.
- activating - starting a tool or opening a drawer by double clicking or by using the OPEN menu option.

Digitize Your Favourite Pics for your Amiga

Any size photograph
slides, etc

\$5 each for 1 to 5 color pics
\$3 each for 5 or more pics

Send blank disk and
money order or cheque to

Daniel Jurisinec
11 Glengate Street
Geelong, Vic, 3215

Send \$10 for a demo disk
Color printouts available on OKI

Icon Types

The .info file stores all the necessary info to display an icon and to startup an application whether the icon represents the tool or a project. The following are the different types of icons:

Icon Name	Object
WBDISK	The root of a disk (main directory)
WBDRAWER	A directory on the disk
WBTOOL	A directly runnable program
WBPROJECT	A data file
WBGARBAGE	The trash can directory
WBKICK	A non-DOS disk

The DiskObject Structure

This makes up most of a .info file. It is defined in C as:

```
struct DiskObject
{
    UWORD do_Magic; /* a unique number to ensure this is a
                    DiskObject */
    UWORD do_Version;
    /* The next member is the actual gadget structure
       including the Image structure which defines the icon
       image */
    struct Gadget do_Gadget;
    UBYTE do_Type; /* type of icon */
    char * do_DefaultTool; /* defines tool to activate */
    char ** do_ToolTypes; /* miscellaneous info for tool */
    LONG do_CurrentX; /* X coord for icon position */
    LONG do_CurrentY; /* Y coord for icon position */
    struct DrawerData * do_DrawerData;
    char * do_ToolWindow; /* specifies default tool window */
    LONG do_StackSize; /* stack size used for tool */
};
```

This structure is defined in workbench/workbench.h and is used for all of the Workbench icon facilities.

Several library routines are available to load, save, allocate and deallocate memory for icons. The routines are GetDiskObject(), PutDiskObject(), and FreeDiskObject(). To use these routines you must open the library icon.library sets up SysBase (Exec's library base pointer) and DOSBase (DOS's library base pointer) and passes the startup message on to main(). When main() returns it replies to the startup message and Workbench then unloads the code from memory. main() is called with two parameters, argc and argv. If argc is NULL the tool was started from workbench, otherwise it was started from the CLI and is the number of arguments entered on the command line in addition to the program name. If called from Workbench the global variable WBenchMsg (defined in AStartup.obj or LStartup.obj) points to the Workbench startup message. Now some detail on the contents of this message before we finish.

Workbench Startup Message

The structure for the startup message (struct WBStartup) is in workbench/startup.h. For our purposes we need only consider the following members of this structure;

sm_NumArgs - The number of arguments in sm_ArgList.
 sm_ArgList - the argument list of icons preselected by the user.
 sm_ToolWindow - the same string as the DiskObjects do_ToolWindow. If not a NULL it is used by the startup code to open a window for the tool. (My testing under version 1.1 shows that this is always a null even if do_ToolWindow contains a window descriptor e.g. "con:40/40/300/100").

Each argument in the list sm_ArgList has two parts wa_Name and wa_Lock. The first argument is the tool's icon even if the tool was activated by double clicking one of its projects (called the default tool in this case). All other arguments are passed in the order the user selected them. Where the argument is not a default tool the wa_Name will be the string displayed under the icon. For a default tool this will be the text of the double clicked project's icon do_DefaultTool pointer. The wa_Lock is a lock on the directory containing the selected icon. An example of code to handle a Workbench startup is:

```
struct WBArg *arg;
char *name;
int i;
LONG olddir;

arg = WBenchMsg->sm_ArgList; /* get argument list */
++arg; /* jump over first argument which is tool */

/* handle rest of arguments passed if any */
for (i=1; i<WBenchMsg->sm_NumArgs; i++, arg++)
{
    /* get file name of project */
    name = arg->wa_Name;
    /* set directory to where this file is */
    olddir = CurrentDir(arg->wa_Lock);
    /* read in DiskObject for .Info file */
    diskobj = (struct DiskObject *)GetDiskObject(name);

    /* ..... */
    /* use tooltype entries to determine type of
       file and if its one of ours and then
       process as required */
    /* ..... */

    /* recover the memory allocated to store the
       DiskObject */
    FreeDiskObject(diskobj);

    /* restore directory */
    CurrentDir(olddir);
}
```

This code fragment is demonstrated in wb_startup.c. Once the DiskObject has been loaded the ToolTypes array can be analyzed with the functions FindToolType() and MatchToolType() if you have formatted your strings as recommended above. The ToolTypes array is used in my example (wb_startup.c) to determine if the icons were of the type config or example. If not of this type the icon is ignored. In wb_startup.c only the first config selected is loaded and all example files are loaded. Normally you would put a limit on the number of files that could be loaded at the one time. My example only loads the icons, and displays the name of the data files in the order they were selected, the actual data files are not loaded.

Well now there's no excuse for not fully supporting the workbench interface in your C programs. Of course you'll need the ROM Kernel Manual to fully understand the DiskObject structure and the startup message, but I hope this article and the demo programs have helped you.

-- Peter Story

Public Domain Software

At the last AUG meeting, I had a member come up to me to complain that a particular program on one of the public domain disks didn't work. After we pushed our way onto an Amiga (hard to do when you are competing with 200 other people!), I found that it was an ABASIC program. When I asked the member if he had ABASIC, the answer was no. After I told him that he couldn't run the program without ABASIC, we got into an "argument" about why we sold disks that people can't run. Here's my explanation.

First of all, public domain software is just that. It is software that, for whatever reason, the author has decided to "give away". Maybe the author didn't want the hassle of marketing it, maybe he was feeling generous. Maybe it was a program that he hacked up to do a small job, and someone told him it would be useful to other people. Maybe the program is a pre-release version of some soon-to-be-marketed program, and the public domain version is a sort of "teaser" to get people interested. In the case of the Amiga, lots of public domain programs the result of programmers "getting used to" the machine.

Unfortunately, some public domain programs do not work properly, contain bugs, or require other programs before they can run. This does not make the release of such a program "wrong". Here's the crunch, in slightly legal terms:

Public domain software is supplied without warranty of any kind. None at all! It may work, and it may not. It may do what it promises, it may even do what you think it might do. Maybe not. Perhaps you will need a compiler to use some of it, perhaps you will need more memory or more disks drives. Maybe, as in the case in hand, you will need something that is not available, ie ABASIC.

We do not sell public domain software. In fact, most public domain software includes a notice that forbids it. We merely charge a small fee to cover our costs in obtaining the disks and copying them. The Fish disks, for instance, cost us over \$12 each by the time we pay airmail and change our dollars into US ones. We, like the original authors, cannot and will not offer any warranty on the programs contained on the disks. If you need something you haven't got to run a program, sorry, but that is **your** problem.

I don't want this note to sound like I don't care. If you pay for a disk and it is no good to you, and you are still upset about it after reading this, then I'll make sure you get your money back.

-- Peter Jetson

First Report from SMAUG (Subgroup Music - Amiga Users Group)

Who am I to ignore the fashion in computing for acronyms (to do so would leave me an anachronism) and how could I resist AUGmenting such a colourful character? I have it on good authority (Ben, age 13) that Smaug was an ancient Red Dragon with at least eleven hit dice and rated extremely intelligent. His connection with music is non-existent but that only goes to make more evident the inscrutability of dragons!

We had our inaugural meeting at 1pm, just before the main meeting on Sunday, Feb 8. Ten people crowded into a little Box Hill lounge room and exchanged interests. I was surprised at the range of backgrounds and directions in such a well defined interest group and delighted at the prospect of future exchange of ideas.

A few more people contacted me after the main meeting or later by phone indicating their wish to become involved so that the group is nothing short of thriving after only one meeting. Clearly there is a great thirst for special interest groups and anyone half thinking of suggesting one should be strongly encouraged.

So far we are these:

Tony Austin & son Mark	- classical to Wakeman, listening to synthetic music.
Tony Major	- using Yamaha FM voices with external Akai keyboard, MIDI driven, looking to learn more.
Mark Webb	- eavesdropper (sounds like a good programmer).
Tim Edgoose	- acoustic guitarist with computing talents, getting into synths and looking for composing tools.
Bob Laidlaw	- (now that I've retired I'm busier than ever) working with Roland & Yamaha keyboards & synths, looking for more software and more possibilities.

Wayne Coles-Janess-	interested in composing and sequencing- and the only one of us who looks remotely like a contemporary pop musician.
Ann & David Peel	- Ann: classical violinist looking for composing aids and scoring facilities; David: clearly an energetic enthusiast, (read his articles in last month's Workbench).
Chris Wood	- guitarist looking for MIDI backing band (my own dream - no more fights with petulant drummers or artistic arguments with bassists).
Greg Freame	- guitarist entering synthetic music.
Nigel Green	- composer with classical training looking for help with film scores.
Dennis Jurisinec	- pianist handy with soldering iron (modestly admitted building a MIDI interface from scratch).
Ian McGill	- experience in keyboard bands and interest in computer graphics, using FM keyboards and MIDI equipment.
Hugh Gurney	- highly expressive classical guitarist looking for scoring software - printing and archiving. Hugh is working on our first software report on music printing.
Shane Powell	- lots of connections with the music industry, looking for experience in synthetic music,
Roland Seidel	- would love to make money out of music -- cure the common cold, find an honest politician and bring peace and happiness to all.

Quite a mixed bunch. Our next meeting will be, once again, just before the main one and fairly close. There is a shabby looking school called Berengarra being slowly overrun by the Hayville Village Retirement Village on the corner of Elgar Road and Canterbury Road in Box Hill. I shall probably get there about 12 noon on the day of the next Users Group meeting and try to place meaningful signs directing you to the music room (if they don't work, it is the North-Western-Most room). Enter on the Elgar Road hill top just south of the intersection.

-- Roland

Demonstration At Next AUG Meeting

SMAUG will be running a demonstration of music with Amigas at the May AUG meeting. If music and the Amiga is a subject you are interested in, make sure you come.

You can contact SMAUG for more information by ringing Roland Seidel on (03) 890 3934 (decent hours only!).

AmigaLink

The Amiga Users Group's new OPUS bulletin board is now online, at 300, 1200 & 1200/75 bps

(03) 792 3918

FidoNet Node # 631/324

AMIGA 2000 and IBM compatability

This message originally appeared on Usenet, a world-wide network of "large" computers, and after reading it, I thought it to be of sufficient interest to reprint it here.

To make it easier to understand, just ignore the first 10 lines or so. They consist of who the message is from, who it is to, where it came from and how it got to where I saw it. In the message proper, lines that start with the ">" character are from the message to which this is a reply. This message has been reformatted to fit in with our newsletter style.

Relay-Version: version B 2.10.3 alpha 4/15/85; site goanna.oz
Path: goanna!munnarilseismolrutgers!cbmvax!daveh
From: daveh@cbmvax.cbm.UUCP (Dave Haynie)
Newsgroups: comp.sys.amiga
Subject: Re: AMIGA 2000 and IBM compatability
Message-ID: <1617@cbmvax.cbm.vax.cbm.UUCP>
Date: 1 Apr 87 23:00:05 GMT
Date-Received: 2 Apr 87 11:37:44 GMT
References: <2845@ecsvax.UUCP>
Organization: Commodore Technology, West Chester, PA
Lines: 153

In article <2845@ecsvax.UUCP>, urjlew@ecsvax.UUCP (Rostyk Lewyckyj) says:

> O.K. folks. This thursday IBM is going to announce its new
> PCs.

All too true.

> According to all the rumours they will use a new
> proprietary buss, and none of the existing expansion boards
> will work in this new machine. Also at the same time or
> shortly later, IBM will be announcing a new operating
> system to go with these new machines. Of course the BIOS of
> the new machines will also be different.

Also true. Best guess is that they'll use a new, Microsoft supplied OS that may depend on proprietary op-codes in special IBM versions of the <ugh>86, <ugh>286, and <ugh>386. Old Messy-DOS or PeePee-DOS will run, but the hardware is supposed to be something completely new.

> So where does this leave the AMIGA 2000, (and the new MAC
> ?) which have been making so much of their their IBM PC
> capabilities? I realize that there is still the world of
> several million existing (old style) IBM PCs, their clones,
> the clone makers, and the add on manufacturers. But the
> fact remains that as far as capturing a piece of the IBM
> market, COMMODORE and all the rest are about to be
> finessed.

Horse-puckey! You're obviously confused on several points.

POINT 1: Pretend for a minute you're a hardware vendor. You're making a new add-on thingy, for the market as it exists today. You could make it for the Amiga Zorro bus (potential market 150K+ users), the Mac II (potential market 0 users), the new IBMs (potential market 0 users), the Commodore 64/128 (potential market 7M+ users), or the PC[clone] market (potential market 10M+ users). Now, personally, I'd like to sell my new hardware thingy. And the best way to sell it is to sell it to the largest available market. That's what PC compatibility on the A2000 gets you, hardware-wise (don't know if the stuff for the Mac is hardware or just software compatible, but the same principals apply). When the new IBM comes out, it'll have the same lack

of add-ons that the Amiga, Mac II, etc. all have. Its a new architecture, and no one's going to build for it until there's an installed base.

POINT 2: CLONES! IBM's getting killed in the PC market. No surprise. Prior to the IBM PC, IBM had it good in the business world. Business zoids bought their typewriters, terminals, and mainframes (they even put up with EPSIDIC characters). Then IBM introduced the PC, a barely-better-than plain-8-bit box they whipped up in 6 months. Who cares, they were IBM, and business zoids always buy IBM, right? Well, at first they did, and IBM made an impact on the business PC world. Then, along came startups like Compaq who built a better PC for less than IBM. And eventually, the business zoids found out that a Compaq, Tandy, Commodore, or maybe even a Hyundai Clone would do the same job, for much less. So IBM started getting killed as far as sales went, in the very market that they created. Every year Clone makers were getting a bigger piece of the IBM market. So now IBM is going to a new architecture, something that "can't be cloned". Big deal. Its Compaq clones from now on, baby!

> The AMIGA 2000 was/is supposed to feed of the availability
> of cheap IBM PC expansion hardware. But with the new
> standards being set by the new machines, how much longer
> will the old expansion addons continue to be available?

The hardware's cheap because there are millions of machines that use it. And folks buy those machines 'cause they use the cheap hardware. Its the story whenever you introduce a new machine. The Clones, now and in the future, all work with the currently available hardware. So does my A2000. But the new IBM beasts don't. So what am I gonna buy next time I need a new PC? If IBM delivers something that much better than my A2000, a Mac SE or Mac II, or a power Compaq or Tandy, maybe some will buy it, and pay major \$\$\$ for the add-ons. And there may be a few holdouts that still want IBM only. But in any case, its IBM that's non-standard now, and everyone knows it.

> Going of on a tangent from the previous paragraphs, I do
> not really understand what if any are the constraints on
> what kind of IBM PC expansion boards will work with the
> AMIGA 2000. Can anybody from Commodore please post such a
> list or description.

I think they've found one '286 add-on board that doesn't work with the XT Bridge Card. No biggie, a '286 Bridge Card is on the way. The only other problem would be adding a second COM port in the normal manner, as the Bridge Card uses the interrupt Messy-DOS dedicates to the second COM port for its Amiga interfacing.

> Does an IBM PC or AT type cpu card have to be installed, in
> order for other IBM PC or AT expansion cards to be useable?

Some kind of Bridge Card is required to access any IBM bus peripherals. Instead of using an XT or AT Bridge Card, you could conceivable develop a dumb bridge card that essentially maps in the PC bus as an Amiga bus device. I do wonder, however, if there's much desire to do this, 'cause then you'd need all kinds of Amiga side software to drive this GREAT STUFF, instead of relying on already existing PC based software.

> Which of the following IBM PC expansion cards and card
> types will/will not work on the AMIGA 2000:
> Multifunction cards e.g. AST Six Pack, 327x emulation cards
> e.g. DCA Irma,
> IBM EGA graphics card, IBM PGA graphics card, IBM PCnetwork
> adapter,

> IBM Token ring adapter, 3M Ethernet card, Above board or
> other memory
> expansions, 3086 accelerator cards, scsi cards.

Most of this stuff should run without trouble on the PC side of things. I know an extensive pile of things has been tested, though I haven't been personally involved in this testing yet. Video cards of any kind will definately work, replacing the Amiga Monochrome/CGA emulation (its a jumper option on the Bridge Card). You could probably even run an Ethernet card if you wanted to, though Amiga bus Ethernet card work so much better.

> By work, I mean that their facilities will be available to
> the AMIGADOS user in some reasonable fashion. That is if
> PCDOS is running is an (intuition?) window, it will be able
> to use say the IRma card and the E78 program to emulate a
> 3278 terminal. Or use the PCnetwork to access files from a
> network server. Notice that I am not asking for these
> services to be available to an AMIGA program, although of
> course if thats possible, I'd like to know about it.

You're only going to be running the PC in an Amiga window if you are using the Bridge Card in its normal mode. Adding a graphics card would certainly remove the need to run in an Amiga window, and in most cases limitations on the IBM side prevent the machine from running with multiple graphics cards (they all vie for the same absolute memory locations in the PC's 1 meg address space, or something like that. I'm not all that attune to such primitive behavior (-:). The Amiga gets access to the PC side though 128K of shared RAM (64K of which is normally video card RAM). When the PC side is coming up, it stops prior to reading PC configuration ROM memory space and notifiys the Amiga side that its ready to read said ROM. This gives the Amiga the opportunity to download driver code to the PC side to do virtually anything. The Amiga can even take complete control of the PC side, to use it as an I/O processor or whatever, independent of MS-DOS. The only limitation would be the shared RAM bottleneck, but of course in such a case the PC processor could pre-process stuff to be sent over to the Amiga side. There's a standard Amiga run-time library that manages the Amiga to PC interface.

> P.S. I have an Amiga, but I am not a true believer, ready
> to defend it, "because it's my computer right or wrong".

I can hardly claim to be non-biased, but I do know just about everything there is to know about the A2000 proper, and I have a reasonable working knowledge of the Bridge Cards.

> However I am not an Ed Chaban, out to villify the AMIGA, or
> to insult you (or myself) by criticising it. I only want
> to evaluate the situation accurately without bias.

That's obvious, I haven't seen any Amiga bashing, half truths, or praises for non-existant hardware from THE OTHER GUYS in this message at all.

Dave Haynie Commodore Technology
{ihnp4|caip|rutgers!}cbmvax!daveh

Commodore rarely admits to knowing me,
much less sharing my personal opinions.

Modula-2 Survey

Due to an oversight while pasting up last month's newsletter, I neglected to include the author's name on the Modula-2 article. Sorry about that. The article was written by Peter G. Evans.

Public Domain Update

As is normal, Fish disks 54 to 58 arrived the same day I took the newsletter to the printers. Those of you who came to the April meeting will already know about these disks. For others, here is the list.

The next lot of Fish disks (59 - 68) are on their way now, but unfortunately this issue of the newsletter will have been already printed by the time they arrive. However, we have downloaded the contents pages of the disks from Usenet so that you'll know about them sooner. Now, no doubt, the problem will be that the disks haven't arrived in time for the meeting! Sigh, you can't win, can you ...

Fish Disk #54

Hanoi	Classical demo program for recursion. Solves the towers of hanoi problem in a workbench window of its own
ISpell	A quick and dirty port of a Unix version of a freely distributable screen oriented, interactive, spelling checker. I use the Unix version daily and it is very nice. You will need expansion ram to run this with the supplied dictionary, as it loads the entire 300K hashed dictionary into memory. A hard disk is also recommended
Ing	The next step in the "boing wars". Turns a nice screen full of little windows into a screen of lots of bouncing little windows
Lav	A "title bar type" program that displays the number of tasks in the Amiga's run queue, averaged over the last minute, 5 minutes, and 15 minutes. Presumably inspired by, and named after, the BSD "load average" program
MidiTools	Simple programs to play and record through the MIDI I/F.
MoreRows	A program to make the workbench screen larger than normal. The number of additional rows and columns are set via command line arguments
Tilt	Another of Leo's cute little toys. This one makes your Amiga look like it didn't pass Commodore's vibration testing

Fish Disk #55

Csh	Version 2.05 of Matt Dillon's csh like shell, modified for Manx C
NewStartups	A couple of new C startup modules. AStartup.asm is the source to AStartup.obj, with 1.2 fixes and better quote handling. TWStartup.asm is like AStartup.asm but opens a stdio window, using a user supplied window specification, when executed from workbench
Palette	A tool which allows you to change another program's custom screen colors. Based on Charlie Heath's palette program on disk 1
PipeDevice	A working 'pipe' device, which allows the standard output of one process to be fed to the standard input of another process, with both processes running concurrently
ScreenSave	A program to save a normal or HAM mode screen as an IFF file. Also creates an icon for it
ShangaiDemo	Demo version of the Activision game "Shanghai".
SoundExample	A double buffered sound example for Manx C using 16-bit ints, small code, and small data
Vsprites	A working vsprite example
Vt100	Version 2.6 of Dave's vt100 terminal emulator

with kermit and xmodem file transfer. It just keeps getting better and better.

Fish Disk #56

Clipboard	Clipboard device interface routines to provide a standard interface, such as Open, Close, Post, Read, Write, etc
ConPackets	Demos the use of DOS packets, finding the Window pointer and ConUnit pointer of the CLI window, toggling Raw mode, getting cursor position and limits from the ConUnit, and ESC-sequence cursor positioning
GetDisks	Sample program to find all available disk device names and return them as a simple exec list. The list is made of named nodes, with the name being the device name
GetVolume	Sample program to get the volume name of the volume that a given file resides on. Works on any device, even the RAM: device
Icon2C	Reads an icon file and writes out a fragment of C code with the icon data structures, for inclusion in a larger program
MergeMem	Program which attempts to merge the MemList entries of sequentially configured ram boards. When successful, allows allocating a section of memory which spans board boundaries
mCAD	An object-oriented drawing program, version 1.1. Uses a small set of graphics primitives (like "line", "box", and "text") and a small set of editing functions (like "move", "size", and "rotate"). While drawing and editing, the user can call on other functions to modify the display; to zoom in, slide around, superimpose a grid, etc. This shareware program was submitted by the author

Fish Disk #57

CutAndPaste	Public domain implementations of the Unix cut and paste commands. Includes source
GraphIt	A program to plot most simple functions in 2 or 3 dimensions, as well as 2d parametric equations in term of t. Includes source
Juggler	Stunning animation of a robot juggler with ray traced reflective spheres. Uses HAM mode display and sound effects to boot! This is version 1.2 and apparently fixes some bugs in the version released on disk number 47
MouseReader	Shareware program, submitted by the author, to read text files and view iff files using only the mouse. Binary only
Ogre	A game of tactical ground combat in the year 2086. Ogres are giat cybernetic tanks, each prodigiously armed and possessing a limited self-awareness, allowing them to do their own tactical planning. Your goal is to neutralize the ogre. Includes source
Splines	Program to demonstrate various curve fitting and rendering techniques. Also includes something unique for the Amiga world, pop-up menus. Includes source

Fish Disk #58

ASDG-rrd	Extremely useful shareware recoverable ram disk. This AmigaDOS device driver implements a completely DOS compatible disk device in memory that survives resets, guru's, and crashes. An absolute must for those with lots of ram. Binary only
----------	---

BigView	Displays any IFF picture, independent of the physical display size, using hardware scroll. Default display size is 320 by 200 in lo-res; HIRES or LACE attributes added if user width/height exceeds low resolution boundaries. Includes source
EGraph	Egraph reads pairs of x and y values from a list of files and draws a formatted graph. Supports four unique curve fonts; solid curves, dashed curves, dotted curves, and long dashed curves. The maximum number of data points is unlimited. Has globs of options. Binary only
HyperBase	Nice little shareware database management system. Version 1.5. Binary only
MemClear	Walks through the free memory lists, zeroing free memory along the way, and coalescing memchunks that have contiguous address spaces. Includes source
NewZAP	A third-generation multi-purpose file sector editing utility, from the author of FileZAP. Displays and edits full 512-byte sectors via a 106 character wide internal font. Includes a search feature to find specific strings or hex digits, forwards or backwards. Version 3.0, includes source
RainBow	Marauder-style rainbow generator. Installs a user copper list such that the background color is changed every few scan lines.
SmusPlayers	Two SMUS players, to play SMUS IFF music formatted files. Executables only
View	A tiny ILBM viewer, for use with either the CLI or WorkBench. Includes source
WBdump	JX-80 optimized workbench printer that does not use DumpRPort. Much more efficient than the Amiga JX-80 driver for full screen dumps.

(NOTE: As mentioned above, Fish disks 59 - 68 were not available at the time of going to press (27-Apr-87), but they will **probably** have arrived by the time you read this. Please check before placing orders for these disks. If you decide to order anyway, we'll send them as soon as we get them.)

Fish Disk #59

Browser	Another version of the browser program released on disks number 18 and number 34. Includes some bug fixes and enhancements
Browser2	This browser type program is apparently not based on the original Mike Meyer version
Clock	Another clock program, comes in several flavors depending upon features desired, which include using alternate fonts, using alternate colors, setting the time, etc
Dme	Version 1.22 of Matt's text editor. Dme is a simple WYSIWYG editor designed for programmers. It is not a WYSIWYG word processor in the traditional sense. Features include arbitrary key mapping, fast scrolling, title-line statistics multiple windows, and ability to iconify windows
DropCloth	Dropcloth replaces the standard blank WorkBench backdrop with a pattern, of setable intensity. Binary only
DropShadow	A program that makes your WorkBench windows have dropshadows. Neat. Binary only
FixWB	A program similar to "DropCloth" (also on this disk), but not fully working yet. At least this one is provided in source, so you get your choice of a working one in binary or a nonworking one in source. Sigh

mCAD	An object-oriented drawing program, version 1.2.2. Uses a small set of graphics primitives (like "line", "box", and "text") and a small set of editing functions (like "move", "size", and "rotate"). While drawing and editing, the user can call on other functions to modify the display; to zoom in, slide around, superimpose a grid, etc. This shareware program was submitted by the author. Many improvements over the version released on disk number 56. Binary only
Robotroff	Another of Leo's cute little display hacks. This one has a definite attraction to pointers (don't want to spoil the surprise)
Supermort	A general compounding/amortization routine, using the intuition environment, which can be used for mortgage/loan computations.

Fish Disk #60

Blitz	Blitz is a small program that is designed to be loaded into memory and that sits in the background until activated by its hotkey. It allows you to view a text file, much like a TYPE command, only that it lets you move forwards and backwards through the file. Its screen updates are blitz'n. Binary only
BlitzFonts	BlitzFonts makes text output up to 6 times faster, transparently to well behaved programs. It is also very small and written 100% in assembly for maximum speed
HandShake	Handshake is a full featured VT52/VT100/VT102 terminal emulator. The author has taken great pains to support the full VT102 spec. This is version 1.20a, binary only
Med	Yet another Amiga text editor. This one lets you edit up to 36 files simultaneously and makes extensive use of the mouse. This is version 2.1, binary only
PrtDrvGen	Program to automatically generate custom printer drivers. Version 1.1, binary only
Show	A nice, very small slideshow type program, version 2.1, binary only
Uedit	Version 2.0 of this nice shareware editor. Has learn mode, a command language, menu customization, and other user configurability and customizability features. Binary only
Ueturbo	Example of extensive customization of Uedit to set up a nice development environment.

Fish Disk #61

ATPatch	A program which reportedly will patch the Amiga Transformer for operation under Workbench 1.2
FillDisk	Disksalv has been known to find some rather interesting things in the free blocks of some production disks from companies that should know better. This little hack makes sure you don't get caught in the same trap, by scribbling the disk's free blocks in a totally safe manner
LPatch	Patch for programs, such as 'Atom', with bad 1.0 Lstartup code, which abort during startup under 1.2 with 00038007 alert (can't open dos library). Includes source
MicroEmacs	Version 3.8b of Daniel Lawrence's variant of Dave Conroy's microemacs. This version is greatly enhanced over the last version, distributed on disk number 22. For example, there is now a full extension language and

PearlFont	A font similar to Topaz, but with smoothed out edges and more rounded characters
Terrain	Program which demonstrates generation of good looking pseudo-random scenery. Includes source in Draco
VSprites	Vsprite example from Rob's book "Programmers' Guide To The Amiga". Produces 28 VSprites onscreen simultaneously, using only three distinct sets of colors. Includes source

Fish Disk #62

This disk contains a port of the popular UNIX game "Hack", done by John Toebe and the crew at the Software Distillery. This is version 1.0.3D

Fish Disk #63

This disk contains a port of the popular UNIX game "Larn", done by Edmund Burnette and the crew at the Software Distillery. This is version 12.0B

Fish Disk #64

This is a copy of the Amiga Developer's IFF disk, received directly from Commodore-Amiga sources, with permission to place in the library and redistribute. It is an update to disk number 16

Fish Disk #65

Bawk	Text processor inspired by the Unix awk utility. Bawk searches files for specific patterns and performs actions for every occurrence of these patterns. The patterns can be regular expressions. The actions are expressed using a subset of the C language. Unfortunately this version always gets a stack overflow no matter what the stack is set to, I haven't had a chance to find the bug. Includes source, so you can hunt for it. Looks like it could be a very useful utility for the Amiga
CloseWB	Simple program for use with MWB (also on this disk) to close a current WorkBench screen, and let you open WorkBench on a custom screen
Cookie	Fortune cookie program. Includes source
JTime	Detailed instructions, including schematics in IFF format, for building and installing a battery backed up real-time clock. The clock goes on the joystick port (aka mouse port 2)
MenuBuilder	A program which automates the process of building menus. It takes a simple text file and generates a C source file with all the needed structures for linking with the rest of your program. This is version 1.0
MWB	A program which will create a new 'WorkBench' screen and route by request OpenWindow calls meant for the WorkBench to these new screens. This allows you to run programs which normally open windows on the WorkBench screen to use a custom screen instead
NewPackets	Tutorial downloaded from BIX C-A support section, which describes some new packets and structures in 1.2 AmigaDos
PascalToC	A Pascal to C translator program which is supposed to correctly handle function, procedure, and most type declarations.

However, this quick and dirty port didn't fare too well on even a simple little Pascal fragment from Software Tools in Pascal. I don't know if the problem is machine dependencies in the code or bugs. Looks like it could be useful with a little more work than I have time to put into it now

Prep Version 2.1 of a Fortran preprocessor called 'prep', an alternative to ratfor. Prep has better macro facilities, a concise shorthand for array and vector statements, all the standard flow control constructs of forth, and is written in generic, portable C (I made no source changes). Includes source

RunBack A program that allows you to start another program which is independent of the CLI window. This is useful to start programs from your Startup-Sequence, load WorkBench, and then close the initial CLI (which could not be closed otherwise)

SunMouse Makes your mouse behave like the Sun Microsystems Sunwindows mouse. You no longer have to 'click' in a window to make it active. Just move the mouse pointer into the window and start typing. Version 1.0

Fish Disk #66

AmScsi Preliminary documentation for a hardware project to build a SCSI controller board. The design does not support DMA or AUTOCONFIG'ing

Asm68k Full featured macro assembler, version 1.0.1, binary only. Well documented

Assigned Same code showing how to find out whether or not a name has been assigned (via the dos ASSIGN command) before using it, thus avoiding the DOS Insert-Disk requester

Dk A little display hack, inspired by Leo's gems. Written in Modula-2, includes source

Flip Seems like Leo's gems have inspired lots of people. This one is quite cute also. Written in assembler, includes source

Foogol Just what you've been waiting for, a foogol cross compiler for your Amiga that generates VAX assembly code. Now you can port all those Amiga foogol programs to your VAX! Seriously, foogol-IV is a tiny Algol like language and this is a compiler for it

Free Free returns the available free bytes on any storage device that AmigaDos sees as a drive. A list of up to six drives is kept by the program and may be cleared or added to at any time by the user. Includes source

MallocTest A malloc/free test program that allocates and frees randomly sized pieces of memory with random lifetimes, and fills them with patterns that can be checked for corruption. Useful for beating on your vendor supplied memory management routines, or possibly as a poorman's memory test program

Melt Another display hack from Leo Schwab, the master himself... Includes source

Nart Another display hack from the master himself... Includes source

Purty Provides an easy way to change some common printer settings via a small window with several gadgets. Binary only

RayTracer A simple ray tracing program. It is capable of depicting up to 150 balls and a plane that is covered with a tiling of any bitmapped

picture. Binary only and sparse documentation

SendPackets Updated versions of the ASendPacket and SendPacket examples from disk number 35. ASendPacket is an example program for sending multiple packets asynchronously to a dos handler, for those interested in implementing programs with asynchronous AmigaDos file I/O. SendPacket is a general purpose subroutine to send AmigaDos packets. Includes source

SnapShot A small utility for dumping screens. This one works like POPCLI and stays dormant until you press Ctrl-Esc. Binary only

TagBBS Version 1.02 of a shareware BBS system.

Fish Disk #67

AmCat Shareware disk cataloging program

AmigaSpell Very nice intuition oriented shareware spelling checker, version 2.0, binary only

Bouncer A 3-D simulation of a bouncing ball written in Creative Solutions' Multi-Forth

Comm Another nice terminal program, Version 1.3

Dux5 Latest version of directory utility which is a descendant of the original dirutil program by Chris Nicotra. Includes source

HexCalc Nice little hex/oct/dec/bin calculator and converter. Binary only

Icons A collection of some icons for general purpose programs and some particular programs. The "documentation" icon is particularly cute

Mandala A mandala graphic program with sound, sort of Eastern music. Binary only

PersMail Demo version of shareware personal/personnel file manager. Includes list processing, capability to run mailing labels, mail merge output feature, and more. Demo version is binary only and limited to input mode. Yet another nice little clock utility that can sit around in your title bar. Lots of options. Version 1.3, binary only

RSLCLock A little graphics demo that shows 16 3D cubes in a 3D space, all being translated, rotated, and drawn on the screen in real time. Binary only, takes over the machine, reboot to recover

RTCCubes Nice little "Wheel of Fortune" type game, written in AmigaBasic

Wheel

Fish Disk #68

To quote the "Read Me" file: This diskette contains the latest Amiga version of MicroGNUEmacs (MG 1b), a small but powerful text editor that also runs on many other computer systems besides the Amiga.

One of MG's major goals is to be compatible with its "cousin" GNU Emacs, so certain features you may have seen in other versions of MicroEmacs may work differently here, or not exist. Hopefully, you'll find the added features MG provides to be worth the trouble it takes to make the switch.

As well as the commands available on *all* systems MG supports, Amiga MG has many Amiga-specific features: the Amiga mouse (with 24 different functions!), Intuition pull-down menus, the Browser (a very nice way to select files), Amiga function keys, a full-screen editing window, and support for using a different text font in the editing window.

Son of Interesting Stuff

Not much has happened this month in the world of the Amiga so I will start off by having a winge about wordprocessors, Commodore and everything.

Last week Commodore announced a price rise on Sidecar from \$1295 to \$1595, this seems just a little confusing to me when I can buy an IBM clone with 640K two drives and a monochrome monitor for \$1200. Commodore gives a \$300 cash rebate for three weeks, then the week after the cash-back offer ends they raise the price on Sidecar by \$300. Maybe they thought everyone who bought a cheap Amiga would also buy a Sidecar a few weeks later and they could get the money back that way. Not that it really matters because Commodore are out of stock of Sidecar and dealers have not been able to get any for about two months.

Since I first got my Amiga I have been looking and hoping for a really good wordprocessor; almost a year later I still have not found one. I am writing this with Textcraft which I prefer over Scribble, but not by a large margin. I read with great interest a review of Textcraft Plus, but when I saw it I was disappointed to find it was just Textcraft with less icons and more pulldown menus, although it does now support multitasking. Then I saw ads for Prowrite, maybe this was the answer I had hoped for, the last I heard on Prowrite was it still had some serious bugs, and would not be worth marketing for some time.

Vizawrite could be what we have all been waiting for, it is advertised as a "desktop publishing wordprocessor for the Amiga" it supports multiple fonts and IFF graphics. Vizawrite should be shipping from England in a month or two, Commodore is the agent for Vizawrite in Australia so expect to see it "real soon now".

Rumour has it that Delux Music from Electronic Arts has been withdrawn from the market in America, apparently there were too many bugs turning up in the program. I have not yet been able to confirm this with the Australian distributors for EA products, but if you own Dmusic it would be worth sending your registration card into EA in the US if you have not already, because if an update does come out this is the ONLY way you will get it from them.

Acquisition, the long rumoured database from Taurus software, was released in Australia three or four weeks ago. Acquisition is a fully programmable database with an integrated wordprocessor and looks like being one of the most powerful pieces of software available for the Amiga.

Micropro's Silent Service, a World War II Submarine simulation, should be available in about three weeks at \$60 to \$70.

-- Fergus Bailey

SOFTWARE ORDER FORM									
Disk numbers :									
Disks supplied by Amiga User Group @ \$10							\$		
Disks supplied by member @ \$2							\$		
Club Use Only							Total \$		
Receipt #: Mailed on: / /									
Mail to: Amiga Users Group, PO Box 48, Boronia, 3155, Victoria.									
Member's Name:									
Address:									

Application for membership of The Amiga Users Group Inc

Membership is \$20 per year. Send your cheque to: Amiga Users Group Inc, PO Box 48, Boronia, 3155

Surname: _____

Details on this side are optional

First name: _____ (no initials)

Year of birth: _____ Do you own an Amiga: _____

Address: _____

Occupation: _____

Postcode: _____

Interests: _____

Phone Number: _____ STD Code: _____

What services would you like AUG to provide: _____

Dealer's Name: _____

Dealer's Address: _____

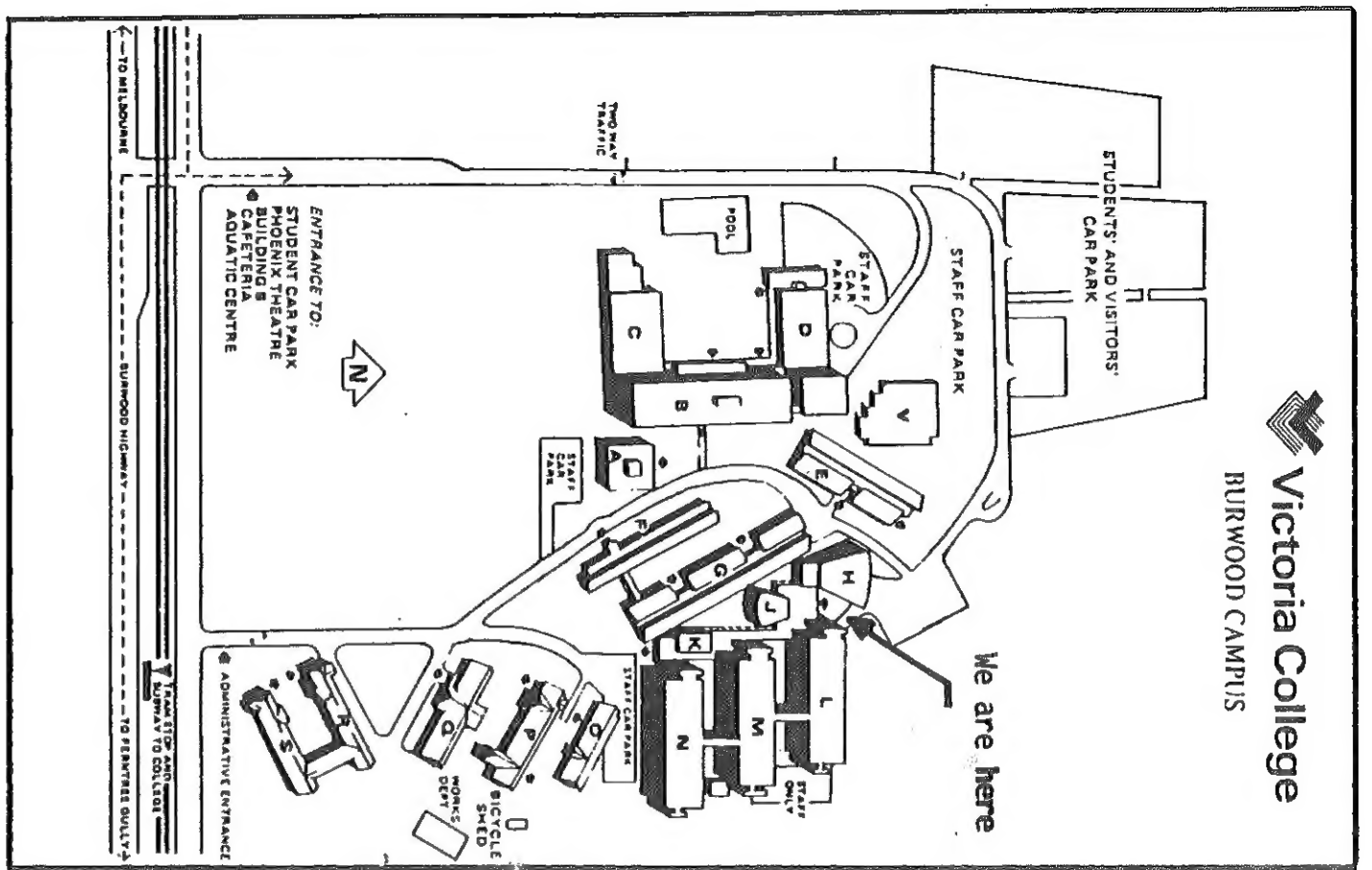
Signed: _____ Date: _____

Postcode: _____

In the event of my admission as a member, I agree to abide by the rules of the Association for the time being in force.

Are you happy with your dealer: _____

Club Use Only	Date	Paid	Rcpt #	Memb #	Card Sent
---------------	------	------	--------	--------	-----------



Where is Victoria College Burwood Campus?

New members and visitors sometimes have trouble locating our meeting place the first time. Victoria College is on the North side of Burwood Highway, Burwood, just East of Elgar Road. Coming from the City, turn left at the first set of traffic lights after Elgar Road. Follow the road around past the football oval, over three or four traffic bumps to the car parking areas near the netball courts. Further up the road, to the left, you'll find Lecture Theatre 2. That's us!

If you have a Melways, try Map 61 B5.

May 1987 Amiga Workbench

P.O. Box 48, Boronia, 3155, Victoria, Australia

AMIGA™ Users Group

AMIGA Workbench
Registered by Australia Post
Publication No: VBG7930
If Undeliverable, return to:
PO Box 48, Boronia, 3155
Victoria, Australia

Postage Paid
Caulfield
East 3145

Member #10297 Expires end of 11-87

Campbell Gray
RMIT Comm Eng Dept
Box 2476V
Melbourne

3001